

(ACCESSION NUMBER)

(THRU)

(CODE)

(CATEGORY)

(NASA CR OR TMX OR AD NUMBER)

Fourth Space Congress
Cocoa Beach, Florida
April 1967

CONFIGURATION MANAGEMENT OF COMPUTER PROGRAMS

Burt H. Liebowitz
Bellcomm, Inc.
Washington, D. C.

H.C. 308
M.F. 50
N67-36770

Abstract

Formal techniques for management control are required to ensure the production and delivery of working, usable computer programs, especially when the programs are elements of large systems. To this end the procedures of configuration management, originally formulated for use in hardware production, have been adapted for use in the production of computer programs. It is the thesis of this paper that these procedures promote more effective management control by requiring:

- a. The development of a two-part computer program specification. The first part defines the requirements of the program; the second part describes the resultant program.
- b. Reviews of the design and implementation of the computer program.
- c. The control of changes to the specification and the resulting computer program.
- d. The maintenance and updating of the computer program specification and other supporting documents.

Introduction

Formally, configuration management is defined as ... "the management of technical requirements which define system, system equipment, or individual equipment, and changes thereto."¹

Informally, configuration management is a tool to prevent a contradiction between what a product is and what its documentation says it should be.

Configuration management was developed to help solve severe problems which had plagued early missile system projects--problems which are characteristic of large scale system development. For example, the lack of precisely defined performance requirements early in some projects led to excessive costs, because many system elements had to be redesigned at a later time to satisfy the users' needs. As the result of another problem--the loss or lack of drawings and technical documentation--

systems were produced that could not be changed or maintained. Even where documentation existed, inadequate change control procedures led to equipments that did not match specifications, and equipments that were incompatible at their interfaces.

Formal techniques for the generation and transmittal of critical information were found to be essential in alleviating these and other similar problems. Configuration management is one such technique. It provides procedures for generating, controlling, and updating the information contained in technical specifications, and for documenting the status of such information.

The policies for configuration management are given in documents such as the Air Force System Command's 375-1,¹ NASA's NPC 500-1² and similar Army and Navy documents.^{3,4} The main purpose of these documents is to provide a basis for the development by contractors and customers alike of consistent procedures for configuration management. They have been successful in this purpose; and it is safe to say that configuration management is here to stay, and will be a factor in more and more large scale government procurements.

Subsequent to the introduction of these manuals there have been some significant changes in technology, not the least of which is the increased use of digital computers as on-line elements in large scale systems. With the use of computers comes the need for computer programs. In many systems the cost of computer programs may be as much or more than the cost of any hardware element. In all cases the programs are critical to the operation of the system. The production of computer programs has thus become a matter of prime concern to those responsible for systems using digital computers.

It is the thesis of this paper that configuration management techniques can be extremely useful in computer programming efforts, primarily because the computer programming process is in many ways similar to the process required for hardware development. On this basis, this paper presents a suggested approach to extending configuration management to cover computer programs.

The procedures summarized in this paper were developed for the Apollo Program Office, NASA Headquarters, for inclusion in NPC 500-1, "The Apollo Configuration Management

Manual."² They are in the process of being incorporated to a limited extent in some areas of NASA.

Background

It is convenient at this point to briefly review for the reader who is new to the subject, some key concepts of configuration management. (The reader who is more familiar with configuration management can proceed directly to the next section.) The interested reader can find the detailed procedures for configuration management in the aforementioned government documents; also, an excellent description of the reasoning behind the development and intent of this tool is given in an article by Laine and Spevak.⁵

Configuration management deals with systems and system elements known as contract end items (CEI's). A CEI is defined as a portion of a system which for technical considerations and managerial convenience is contracted for and produced as a single item. For a computer-based system the CEI's may be such things as electronic equipments, digital computers, computer programs and facilities (see Figure 1).

The overriding concept of configuration management is the orderly development of requirements, first for the system as a whole and then in more detail for its constituent CEI's. The requirements are developed in phases and published in "baseline" documents.

A baseline is a collection of information, as known up to a particular instant of time, describing the technical characteristics--performance, design, configuration, testing--of a system or system element. Baselines are established at discrete times in the life of a system as references for the evaluation of proposed changes to the technical characteristics. The totality of baselines plus approved changes provides an up to date description of the system.

Four phases and three baselines have been defined for use in configuration management (see Figure 2). These are:

The conceptual phase which begins with the realization that a need exists--a need that can be satisfied by a new system. The requirements for the system are defined and a system concept is developed. The requirements are documented in a system specification which becomes the initial, system requirements baseline for the development of the system.

The definition phase in which the structure of the system is defined and the CEI's that comprise the structure are identified. In this phase the performance requirements are generated for

each CEI and documented as part 1 of the CEI specification. Contractors are then chosen to produce each end item. For each end item, part 1 of the specification is established as the design requirements baseline used for controlling the contractor's activities in the following phases.

The acquisition phase in which the end items are designed, developed, produced, and documented. The design and actual configuration of each CEI is defined in part 2 of the specification. This becomes the product configuration baseline; it is used to control changes to the CEI and to control the subsequent production of similar articles.

The operation phase in which the end items are integrated to form the system, and during which the system is used to perform its operational mission.

The controlled development of these baselines provides a framework for documenting the key information required for the system's development as it becomes known. It also allows the user to employ the services of several contractors and still have the various end items compatible when they are integrated.

Thus configuration management provides a basis for identifying, controlling and documenting the configuration of a system and its constituent CEI's. The remainder of this paper will consider those aspects of configuration management that pertain to computer program contract end items (CPCEI's).

The Computer Program Contract End Item

What is this thing we call a CPCEI and how is it produced?

A computer program is the ordered set of instructions and data required to control the operation and determine the outputs of a digital computer. The end product of the process required to produce a program is usually a punched deck of cards, magnetic tapes, or other physical media containing the ordered set in a form suitable for insertion into a digital computer. A computer program contract end item (CPCEI) is an end product of a programming effort, specifically, an end product which has been designated as a contract end item for the purposes of management control.

The end product - the CPCEI - is an entity physically different than the documents which describe it (much like a rocket engine is different from the documents that describe it). The sum total of activities required to produce the CPCEI is called the "programming process." Like the process for hardware end items, the programming process can be considered to take place in a definition, acquisition, and operation phase.

Indeed the types of activities within each phase are somewhat similar; that is:

1. In the definition phase the requirements for the CPCEI are defined.
2. In the acquisition phase the CPCEI is designed, documented, coded, and then tested in a simulated environment.
3. In the operation phase the CPCEI is checked in a system environment, integrated with the other end items, updated as required, formally accepted, and used.

Since the processes are similar, one may conclude that many of the techniques developed for hardware can be applied to the management of computer programming. This is particularly true for configuration management, since many of the problems - loss of documentation, lack of change control, etc. - which prompted its development arise in computer programming.

There are some differences, however, between hardware and computer programs which have to be considered when extending configuration management procedures to cover computer programs. For example, the manufacture of the physical product is not usually a major factor for computer programs while it may be for hardware. Also, production of succeeding articles is not a major factor, reproduction being a simple matter of duplicating tapes and card decks on computing and accounting machines. Also, reliability and quality concepts differ greatly between hardware and computer program items. Computer instructions do not "wear out." Therefore, designing for longevity and provisioning spare parts are not important factors for computer programs. Perhaps the most troublesome difference arises from the "softness" of a CPCEI. Neither its configuration nor its ability to function can be readily determined by visual inspection. You can't kick it, inspect it, or watch it move; things which may have meaning for hardware items. Therefore, a lot of emphasis must be placed on descriptive documents and performance tests to make these determinations.

Configuration Management in the Programming Process

The proposed method of meshing configuration management procedures with the activities of the programming process is illustrated in Figure 3. This method takes into consideration the differences mentioned above but strives to maintain for computer programs a management approach that is similar to that used for hardware.

In the discussion of this method to follow, the group managing the total system will be called the "user," and the organization producing the CPCEI will be called

the "contractor." The term contractor is employed only in deference to the origin of configuration management as a tool to assist government agencies in managing contracted efforts. Configuration management can also be of great value in non-contracted "in-house" efforts.

The starting point for the discussion is the definition phase, just subsequent to the issuance of the system specification. We will assume a system which contains many diverse elements. A single contractor is responsible for the CPCEI; other contractors are responsible for the other end items in the system, and for the integration of the system. It is also assumed that the program is designed and implemented in the programming contractor's facility, and then transferred to the operational facility for integration into the system.

One other point. In the following discussion little or no mention is given to other management control functions or tools such as reports, PERT charts, quality assurance, testing, etc. This is not meant to imply that configuration management is all that is needed for management control; it is rather a reflection of the fact that this is a paper on configuration management.

The Definition Phase

In the definition phase, tradeoff studies are performed by the user to determine the functions of the hardware and computer program elements in the system. These studies are based on the requirements as stated in the system specification and an evaluation of the computer resources available to meet these requirements. Two events in the phase have major significance for the configuration management of each identified CPCEI:

- a. the production and release of the first part of the CPCEI specification;
- b. and the assignment of identification numbers to the CPCEI, its parts, and its specification.

Part 1 of the Specification - The first part of the specification contains performance requirements, design requirements, and test requirements for the CPCEI. The performance requirements describe what the program must do; i.e., for each major function of the program, the data inputs, the required computer processing, and the required computer outputs are given. The design requirements delineate items not directly related to the desired performance of the program, which constrain the design of the program. These include interfaces with equipments and other computer programs, requirements to use available equipments and programs, and required design attributes such as modularity and expandability. The test requirements define the types of tests needed to verify that the CPCEI satisfies the performance and design requirements contained in the specification.

The part 1 specification for a CPCEI differs somewhat in content from the hardware part 1 "spec."* For example it does not contain reliability, transportability, and maintainability requirements. Since there is no great wealth of mil standards for computer program production, there is almost no reference to them in the spec. Also, many of the terms and phrases that abound in a CPCEI spec are unique to computer programs, and may confuse the uninitiated. All this is to be expected since a computer program CEI is a different animal than a hardware CEI.

However, the part 1 specification is very similar in function to the hardware part 1 spec. Upon release or approval by the user it becomes the design requirements baseline for the CPCEI. Once so established, part 1 is used by the contractor as the basis for program design. It functions for the user as a standard by which the performance of the resultant CPCEI can be evaluated, and as the basis for controlling changes to the CPCEI.

Since the released part 1 spec is a baseline document, changes to it must be effectively controlled. This is accomplished by a change control board (CCB) composed of user personnel. The functions of this board and the mechanism of change control will be discussed in more detail later in this paper; suffice it to say now that the CCB must approve all changes to the released spec before they can be implemented by the programming contractor.

The user's approval of the part 1 spec terminates the definition phase.

Identification Numbers - Special numbers are used to uniquely identify the CPCEI, its parts, and its specifications throughout their life cycle. Such identification allows the contractor and the user to relate the tapes and card decks that contain the program to the documents that describe the program. This is a simple point but one that helps prevent loss of these items and confusion between their different versions. Identification starts in the definition phase with the assignment of numbers to the CPCEI and the part 1 spec. It continues in the acquisition phase during which numbers are assigned to the physical parts -- tapes, card decks, cannisters -- of the CPCEI.

The Acquisition Phase

The acquisition phase can for convenience be divided into two subphases: the design subphase, and the implementation subphase. In the design subphase the programming

*A detailed description of the format for and content of a part 1 CPCEI specification is given in Exhibit XVIII of NPC 500-1. This exhibit was published in August 1966, as an addendum to 500-1. It is almost identical to proposed Exhibit XX to 375-1. The description is also included in references (6), (7).

contractor designs the computer program. Concurrently, the part 1 spec may be modified or revised due to additional analysis of requirements and/or feedback from the design process. In the implementation subphase, the contractor implements the design, and tests the resultant CPCEI in a simulated environment; he also produces part 2 of the specification.

The intent of configuration management in the acquisition phase is to ensure that: (1) the CPCEI satisfies the requirements of the part 1 spec; (2) the completed CPCEI is accurately documented in the part 2 spec; (3) the components and parts of the CPCEI are marked for easy identification. This is accomplished, much as it is for hardware, by means of change control procedures, and by means of reviews and inspections at critical times throughout the phase.

Design Reviews - The PDR (preliminary design review), held in the design subphase, is the first formal review. It is held when the contractor's design activity has reached the point where he has allocated functions to individual computer program components (CPC's), and produced flow charts showing the data flow between the CPC's. The PDR gives the user an opportunity to determine early in the programming process if the contractor is designing a product that actually satisfies the requirements in part 1. During the PDR both the compatibility of the selected design approach with the part 1 spec, and the interface compatibility of the CPCEI with the other end items are checked. The output of the review is either concurrence by the user in the design approach, or a set of action items to be acted on by the contractor.

The approved design approach serves as the basis for the detailed design of each identified computer program component. The designs are documented in programming specifications. A programming spec describes the inputs, outputs, and computational methods for a CPC or convenient group of CPC's in sufficient detail to enable a programmer to code and debug that portion of the CPCEI.

Each CPC design is then reviewed by the user in a critical design review (CDR). The prime goal of the CDR is to establish concurrence between the user and the contractor in the design of the CPC(s); this is to be done before significant resources are committed to its development. During a CDR the design of the CPC is reviewed to determine if the completed CPC will satisfy the functions allocated to it; also the interface compatibility of the CPC with the other CPC's is checked. The result of a CDR is either approval by the user of the CPC design or a set of action items requiring redesign effort by the contractor.

For a large program, the individual CPC's may be designed over a period of time, therefore, several CDR's may be necessary. It is the responsibility of the user to schedule the CDR's so that the contractor's design activities can be expedited; to this end several CPC's may be reviewed at a single CDR.

After approval at the CDR's, the individual components are coded, debugged and integrated. These tasks, which take place in the implementation subphase, may take considerable time in a large programming effort. During this time the contractor is free to make changes in the design, provided they do not affect the requirements as stated in the part 1 spec. If they do, the changes must be approved by the CCB. In turn, any changes to the spec generated by the user must be passed on to the contractor by the CCB. This provides the basis for keeping the evolving CPCEI in step with the evolving part 1 spec.

Part 2 of the Specification - The second part of the specification is produced during the acquisition phase. In many ways it is similar in function to the hardware part 2 specification. It describes the exact configuration of the end product, for computer programs the ordered sequence of instructions and data. Once released it becomes the product configuration baseline, used as the basis for controlling changes to the CPCEI in the operation phase, and as an instrument for modifying and updating the CPCEI.

In some ways, however, it is unlike the hardware part 2 spec. For example, for hardware the part 2 spec calls out drawings. For computer programs, however, the spec contains program listings. The listings document the instructions and data. In general, they are produced after the program has been coded and key-punched, as a by-product of the processes known as assembly and compilation. The listings are usually generated by the computer and are the most valid representation of the program as actually used by the computer. Therefore, the part 2 spec for computer programs is a description of the program as built, not a "build to" spec as it is in many hardware procurements. Its primary use is for controlling changes and modifications to an existing program, not for controlling the production of a new copy of an existing item.

Actually, the part 2 spec is a collection of information that is generated during the acquisition phase of any well managed programming effort. It contains in addition to the listings: (1) a description of the overall program design including high level flow charts; (2) programming specifications for the individual components of the CPCEI; and (3), the detailed flow charts for the components. The additional information is included to give more insight into the workings of the CPCEI than could be derived from the listings alone.

The relationship between key activities in the acquisition phase and the component documents that make up the part 2 spec is illustrated in Figure 4.* It can be seen from the figure that many of these documents, in preliminary form, can be used by the contractor as inputs to the design reviews. In fact one way the user can be sure of getting an adequately documented CPCEI is to require the contractor to submit relevant portions of the part 2 spec as inputs to the design reviews. This in itself is a major benefit derived from applying configuration management to programming tasks.

First Article Configuration Inspection - The implementation subphase ends when the CPCEI has been totally assembled and tested in a simulated environment. The CPCEI is now ready to be transferred to its operational environment for integration with the other end items in the system. This is a moment of truth, and sometimes the truth hurts. The shipment of unmarked and poorly documented computer programs, not an uncommon practice, can cause grave problems in the operation phase. For example, to meet a schedule commitment, a computer program may be shipped to an operational facility on a certain date even if it has not been completely checked out or even completely written. The contractor must then complete the program while it is being integrated into the system. Yet he may not have adequate documentation; he can not be sure if errors are the fault of the program or the system; and, if a geographical movement was involved, may not have the key personnel necessary to complete the program. The chances of completing the program without seriously hindering the system's integration are small indeed.

This problem can be avoided by careful scheduling, and by applying discipline to overcome the temptation to transfer an incomplete product. Configuration management supplies some of this discipline by requiring that an inspection be held prior to the transfer of the end item. This inspection for hardware items is called First Article Configuration Inspection (FACI), and for the purpose of conformity the same term will be used for computer programs. The FACI in hardware development is the inspection of the first of a series of articles to determine: if its configuration agrees with its description in the part 2 spec; if it is properly marked; and if all supporting documentation is up to date.

*A complete description of a suggested format and content of a part 2 spec is given in Exhibit XVIII of NPC 500-1 and in references (6), (7).

Although the reproduction of copies of a CPCEI is usually not a significant activity, a FACI is an appropriate and necessary milestone in the programming process. A well run FACI can help eliminate many of the problems associated with the transfer of a computer program from a developmental environment to an operational environment.

In general, FACI is conducted when both the part 2 spec and the preliminary qualification tests have been completed. If the CPCEI is developed at the contractor's facility for subsequent shipment to the user's facility, FACI should be conducted prior to this shipment. If the CPCEI is developed at the user's facility, FACI should be conducted prior to final qualification testing of the computer-based system. The following is accomplished as a part of the FACI for a CPCEI:

1. The existence and proper marking of each program component and each CPCEI part (tape, card deck, etc.) is verified.
2. The completeness of part 2 is verified by visual inspection. When approved, the spec is released as the product configuration baseline. It is used in the operation phase as an instrument for controlling and making changes to the CPCEI.
3. The results of testing the CPCEI in a simulated environment are audited to determine if such tests have been satisfactorily completed.

The result of FACI is either the approval of the part 2 spec as the product configuration baseline and approval to transfer the CPCEI to the operational facility, or a set of action items that must be acted on by the contractor to obtain these approvals.

The Operational Phase

If all is well the specification will accurately describe the CPCEI when both are transferred to the operational facility. The object of configuration management in the operation phase is to keep it that way. In many programming efforts this may not be an easy task. Hundreds and thousands of changes may have to be made to the CPCEI as errors are uncovered during the integration tests. The contractor must be allowed to make these changes as expeditiously as possible, yet it is essential that the specification be kept up to date and that all affected parties be notified of the changes. To accomplish this requires somewhat of a juggling act. Procedures must be developed that are stringent enough to ensure that all changes are reported, but not so severe as to impede the contractor's ability to make the changes. Some procedures for achieving this delicate balance are discussed below. They fall under the general categories of change control and configuration accounting, activities which take place during both the acquisition and operation phases.

Change Control

The term "change" refers to any alteration to an established baseline. In order to establish change control the user must: (1) identify which types of changes are considered critical; (2) require the contractor to submit a formal engineering proposal for each critical change; (3) establish a change control board (CCB) to evaluate each critical change.

The CCB, composed of user personnel, must disseminate to all affected parties the results of its evaluations. In a large system there may be several CCB's arranged in a hierarchical order. A single CCB may have responsibility for several end items. The arrangement of CCB's must be determined by the user prior to entering the acquisition phase for the system. Although many CCB's may exist in a system, the programming contractor should be "aware" of only one, the one that has cognizance in his area. A possible CCB arrangement is shown in Figure 5.

In general, there are two types of changes that are considered by a CCB:

Class I changes - those which affect performance and safety as defined in the established baselines, the cost or delivery date of the CPCEI, or interfaces with other end items.

Class II changes - those which are not Class I (e.g., simple error corrections).

All Class I changes require CCB approval before they can be implemented. Class II changes do not require CCB approval. The initial determination of what is a Class I change is left to the contractor under the cognizance of a user representative. The CCB, however, reserves the right to make the final determination of all classifications.

The criteria for determining if a change is Class I or Class II are derived from ANA Bulletin 445⁸ and are very similar to those used in hardware change control. There is one major difference however. Under strict interpretation of ANA Bulletin 445, a computer program change of any type in the operation phase would be classified as a Class I change, because the listings in the released part 2 spec must also be changed. Since the number of error changes can be large, the contractor must be given leeway in making them without having to wait for the approval of the procuring agency. To avoid unnecessary delays, these types of changes may be considered as Class II even though the part 2 spec must be altered.

It is important to note that all approved changes - Class I or Class II - must be reflected in the pertinent documents, this being a contractor responsibility. It is also important to realize, that until a baseline is formally established by user-contractor concurrence, changes to it are not subject to

formal change control. For instance a contractor is free to make changes to a CPCEI and/or the part 2 spec prior to FACI without CCB approval, as long as the design requirements baseline is not affected. This provides the contractor the flexibility to respond to the essentially iterative nature of the acquisition phase.

Configuration Accounting

Another major activity which continues throughout the acquisition and operation phases is configuration accounting -- the reporting and documenting of all proposed changes to an established baseline configuration. Accounting procedures insure that the baseline is maintained in an accurate and timely manner.

Accurate records must be developed, maintained, and disseminated to assure that the exact status of the CPCEI and its specifications is well known. These records include:

1. Specification change notices (SCN) which record changes to all approved specifications.
2. A specification change log which formally records all approved SCN's.
3. An end item configuration chart which identifies approved engineering change proposals (ECP's) incorporated in a particular version of a specification.
4. The configuration index which provides an official listing of the CPCEI specification and significant support documents.
5. The change status report which is used to provide the contractor and the procuring agency with a summary of the status of all ECP's.
6. The version description document which identifies the elements (tapes, card decks, etc.) of a released computer program.

A detailed description of these documents is given in reference (7).

Concluding Remarks

There are certain basic benefits that could accrue from extending configuration management to computer programs. The contractor would be provided with a detailed specification to work from. He would be assured of receiving notification of all significant changes to the system. The design reviews would tend to minimize misunderstandings as to what the final product must do. There would be less chance

of misplacing or losing the critical documentation that describes the program. There would be a check provided against transporting an incomplete product from a developer's facility. The contractor would be obliged to document the resultant program, and the user obliged to recognize the cost of such documentation. The user would be somewhat protected against contractor generated changes that may run up the cost of a program.

In addition there would be certain other benefits which are less direct but equally important:

- a. The user would be encouraged to consider the requirements of and the resources for computer programs early in the system's development, a necessary prerequisite for effective planning.
- b. The computer program elements of a computer-based system would be placed under the same degree of control as the hardware elements.
- c. Several convenient milestones would be provided for measuring the progress of a programming task. For example, the completion of the design reviews and the release of the two parts of the specification are measurable and definable. Furthermore, by having common milestones, the user can more effectively monitor the progress of a multi-program effort.

It should be noted that many of the benefits just mentioned are not limited to programs that operate as part of large complex systems. They also apply to compilers, scientific application programs, payroll programs, etc. - programs that operate within standard commercial computer facilities. For these types of programs there is a decreased need for a system specification and interface control. However, the other facets of configuration management - baselines, change control, configuration accounting procedures, identification numbers - are essential to their management if they are large enough to require the interchange of documented information.

References

1. AFSCM 375-1, "Configuration Management During the Definition and Acquisition Phases," U. S. Air Force Systems Command, June 1, 1964.
2. NPC 500-1, "Apollo Configuration Management Manual," National Aeronautics and Space Administration, May 18, 1964.

3. U. S. Army Material Command Regulation 11-26, June 1965.
4. U. S. Navy Nav Mat 5200.20.
5. Laine, M. J., and Spevak, F. C. "Configuration Management," Space/Aeronautics, November 1966, p. 74.
6. Liebowitz, B. H., Parker, E. B., III, and Sherrerd, C. S.; "Procedures for Management Control of Computer Programming In Apollo," TR-66-320-2, Bellcomm, Inc., September 28, 1966.
7. ESD Exhibit EST-1, "Configuration Management Exhibit for Computer Programs," U. S. Air Force Systems Command, L. B. Hanscom Field, Bedford, Massachusetts, May 1966.
8. ANA Bulletin No. 445, "Engineering Changes to Weapons, Systems, Equipments, and Facilities," July 12, 1963.

NOTE: THE SYSTEM
DEPICTED IN THIS FIGURE
IS A HYPOTHETICAL SPACE
TRACKING AND COMMUNICATIONS
SYSTEM

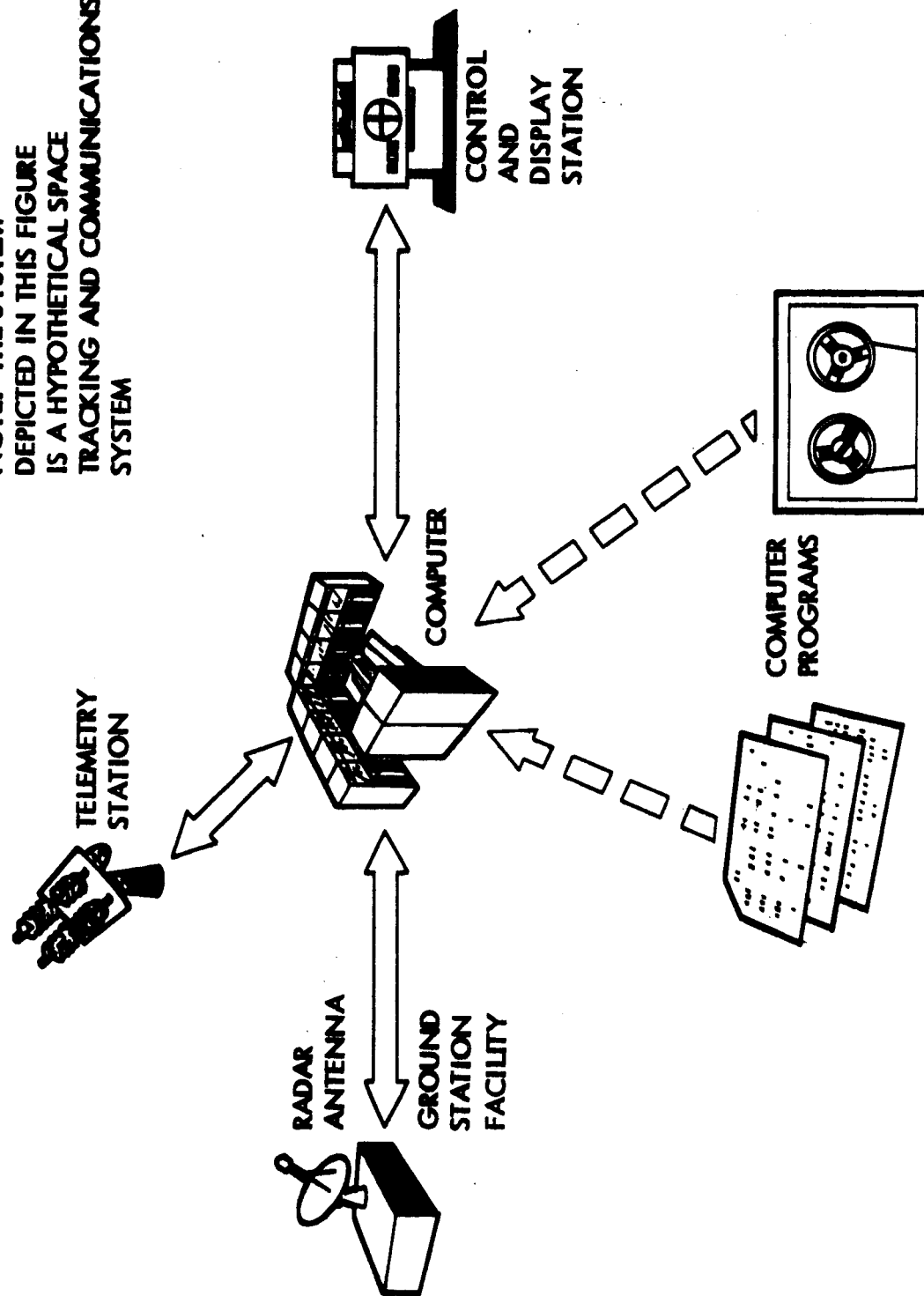


FIGURE 1
END ITEMS WITHIN A
COMPUTER-BASED SYSTEM

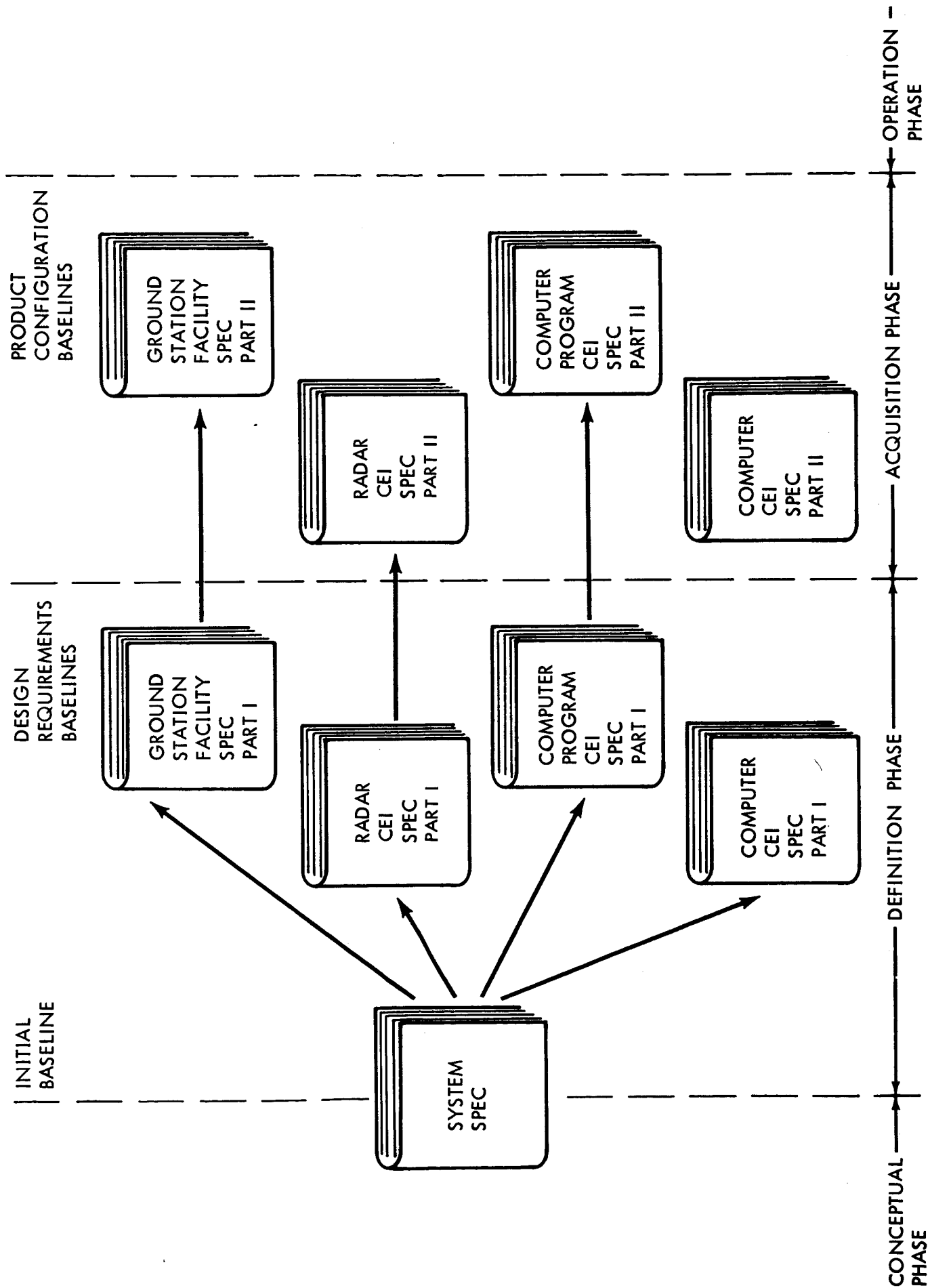


FIGURE 2 BASELINE STRUCTURE

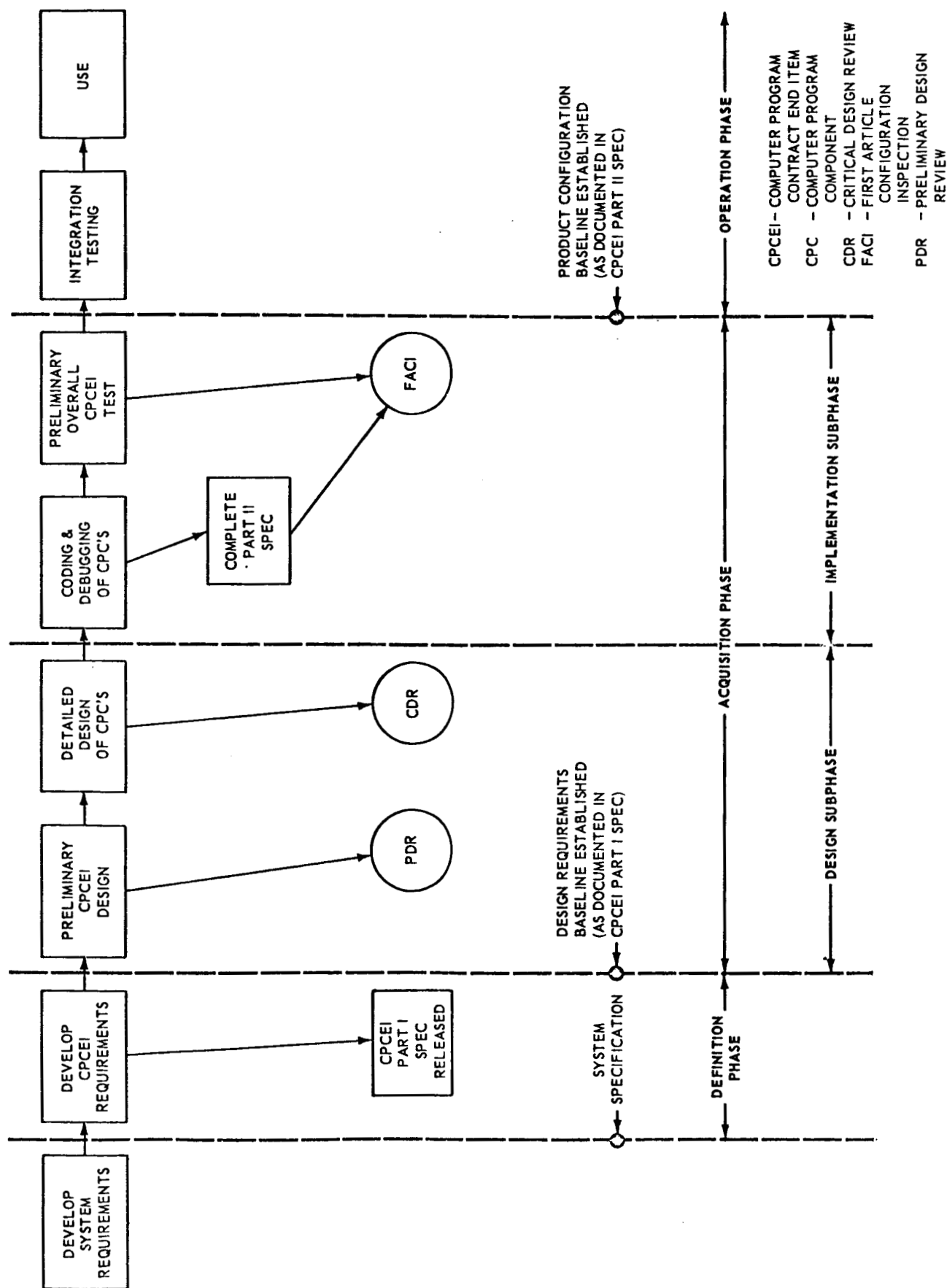


FIGURE 3 MAJOR MILESTONES-COMPUTER PROGRAM CONFIGURATION MANAGEMENT

FIG 4
THE DEVELOPMENT OF THE
PART 2 SPEC IN THE ACQUISITION
PHASE

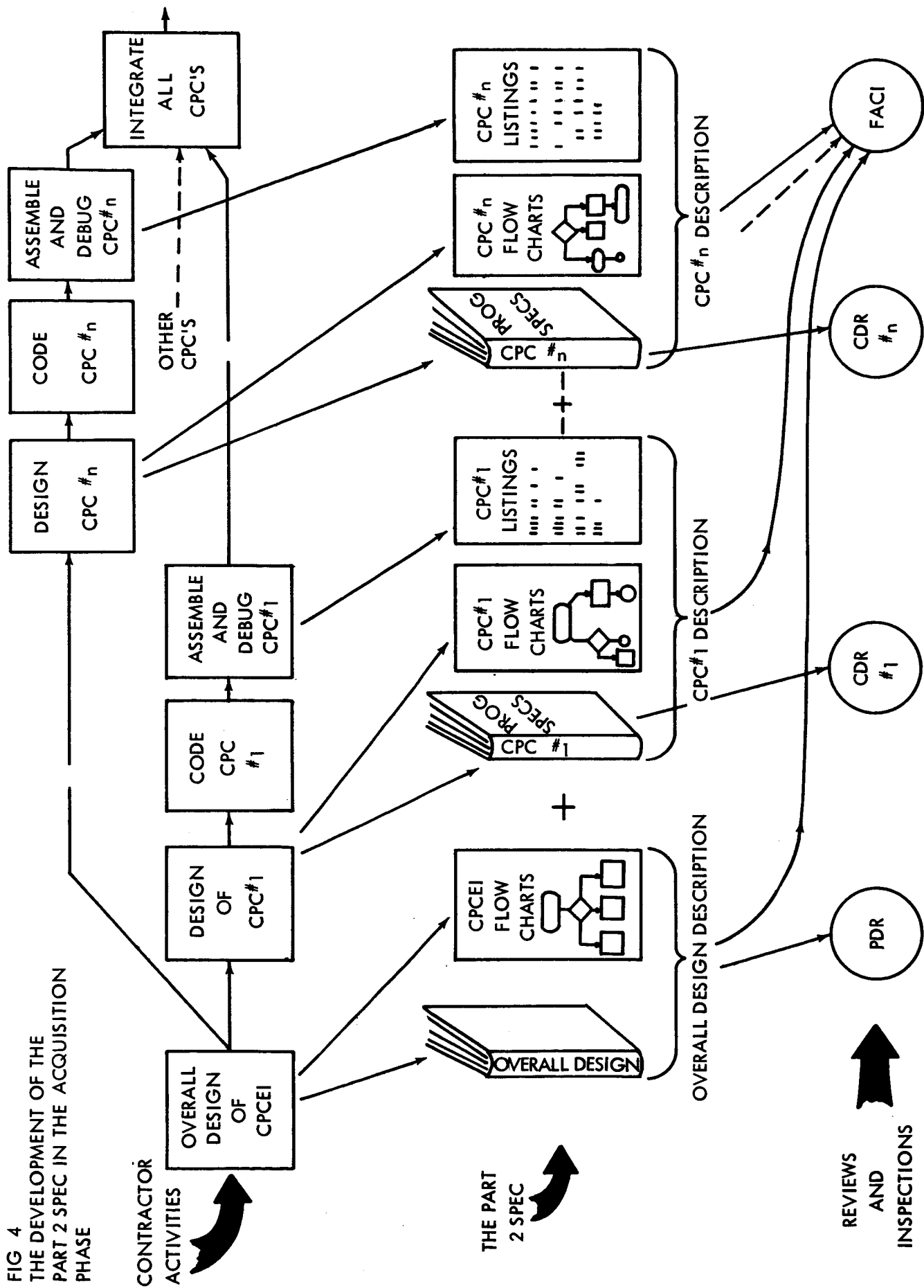


FIGURE 5
A POSSIBLE
CHANGE CONTROL BOARD
ARRANGEMENT

